METHOD AND SYSTEM FOR DETECTING FREQUENTLY APPEARING PATTERN

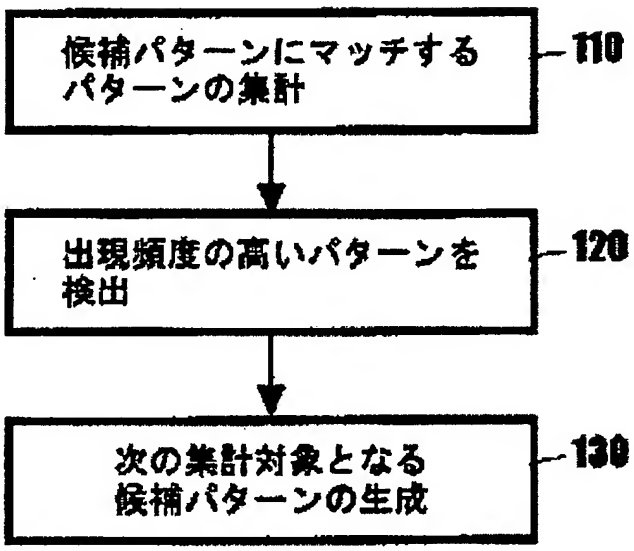
Publication number:	JP2001134575
Publication date:	2001-05-18
Inventor:	FUKUDA TSUYOSHI; MATSUZAWA YASUSHI
Applicant:	IBM
Classification:	
- international:	*G06F17/30;* **G06F17/30**; (IPC1-7): G06F17/30
- European:	G06F17/30T5S
Application number:	JP19990309582 19991029
Priority number(s):	JP19990309582 19991029

Also published as:

US6618725 (B1)

Abstract of **JP2001134575**

PROBLEM TO BE SOLVED: To provide a method and a system for detecting an important pattern included in a set of data. SOLUTION: This system which detects a frequently appearing pattern in a database including a data set represented with tree structure data by using candidate patterns to be totalized has (1) a means for totalizing patterns matching candidate patterns from the database, (2) a means for detecting a pattern of high appearance frequency through the mentioned totalization, and (3) a means for generating a candidate pattern to be totalized next time from the detected pattern.



本発明の概要

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（A）　(11)特許出願公開番号
特開2001－134575
(P2001－134575A)
(43)公開日　平成13年5月18日(2001.5.18)

(51)Int.Cl.'	識別記号	FI		テーマコード(参考)
G06F 17/30		G06F 15/401	320Z	5B075
			340Z	
		15/419	310	

審査請求 有　請求項の数6　OL　（全 19 頁）

(21)出願番号　特願平11－309582

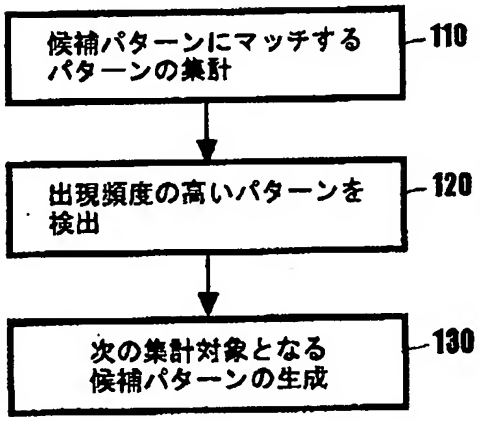
(22)出願日　平成11年10月29日(1999.10.29)

(71)出願人　390009531
インターナショナル・ビジネス・マシーンズ・コーポレーション
INTERNATIONAL BUSINESS MASCHINES CORPORATION
アメリカ合衆国10504、ニューヨーク州
アーモンク　（番地なし）
(74)代理人　100086243
弁理士　坂口　博　（外1名）

最終頁に続く

(54)【発明の名称】　頻出パターン検出方法およびシステム

(57)【要約】
【課題】データの集合からその中に含まれる重要なパターンを検出する方法及びシステムを提供することである。
【解決手段】木構造データで表わされたデータ集合を含むデータベースから、集計対象となる候補パターンを用いて、頻出パターンを検出するシステムであって、
（１）データベースから候補パターンにマッチするパターンを集計する手段と、（２）前記集計により出現頻度の高いパターンを検出する手段と、（３）前記検出したパターンから、次の集計対象となる候補パターンを生成する手段と、を有するように構成する。



本発明の概要

1

【特許請求の範囲】

【請求項１】木構造データで表わされたデータ集合を含むデータベースから、頻出パターンを検出するシステムであって、（１）データベースから候補パターンにマッチするパターンを集計する手段と、（２）前記集計により出現頻度の高いパターンを検出する手段と、（３）前記検出したパターンから、次の集計対象となる候補パターンを生成する手段、を有する、頻出パターン検出システム。

【請求項２】大量のテキストデータから、有用な概念を抽出する、テキストマイニング・システムであって、（１）各テキストデータ中の文章を構文解析する手段と、（２）前記構文解析の結果に基づき構文木を生成する手段と、（３）前記構文木の集合からなる、データベースを作成する手段と、（４）データベースから集計対象となる候補パターンを集計する手段と、（５）前記集計により出現頻度の高いパターンを検出する手段と、（６）前記検出したパターンから、次の集計対象となる候補パターンを生成する手段、を有する、テキストマイニング・システム。

【請求項３】大量のテキストから、ＦＡＱを自動的に作成する、ＦＡＱ作成システムであって、（１）各テキストデータ中の文章を構文解析する手段と、（２）前記構文解析の結果に基づき構文木を生成する手段と、（３）前記構文木の集合からなる、データベースを作成する手段と、（４）データベースから集計対象となる候補パターンを集計する手段と、（５）前記集計により出現頻度の高いパターンを検出する手段と、（６）前記検出したパターンから、次の集計対象となる候補パターンを生成する手段と、（７）前記出現頻度の高いパターンにマッチする、元のテキストデータを抽出する手段、を有する、ＦＡＱ作成システム。

【請求項４】ユーザが複数のカテゴリに分類したテキストデータに基づき、大量のテキストデータを自動的に分類する、テキスト分類システムであって、（１）前記ユーザが分類したテキストデータ中の文章を、カテゴリ別に構文解析する手段と、（２）前記構文解析の結果に基づき構文木を生成する手段と、（３）前記構文木の集合からなる、データベースをカテゴリ別に作成する手段と、（４）データベースから集計対象となる候補パターンを、カテゴリ別に集計する手段と、（５）前記集計により出現頻度の高いパターンを、カテゴリ別に検出する手段と、（６）前記検出したパターンから、次の集計対象となる候補パターンを生成する手段と、（７）大量のテキストデータから、カテゴリ別に検出された前記出現頻度の高いパターンにマッチするテキストデータを検出することにより、大量のテキストデータをカテゴリ別に分類する手段、を有する、テキスト分類システム。

【請求項５】木構造データで表わされたデータ集合を含むデータベースから、頻出パターンを検出する方法であって、（１）データベースから候補パターンにマッチするパターンを集計する段階と、（２）前記集計により出現頻度の高いパターンを検出する段階と、（３）前記検出したパターンから、次の集計対象となる候補パターンを生成する段階、を有する、頻出パターン検出方法。

2

【請求項６】木構造データで表わされたデータ集合を含むデータベースから、頻出パターンを検出するプログラムを含む記録媒体であって、該プログラムがコンピュータに、（１）データベースから候補パターンにマッチするパターンを集計する機能と、（２）前記集計により出現頻度の高いパターンを検出する機能と、（３）前記検出したパターンから、次の集計対象となる候補パターンを生成する機能、を実現させる、プログラムを含む記録媒体。

【発明の詳細な説明】

【０００１】

【産業上の利用分野】本発明は、複雑な構造をもつデータ集合から、重要なパターンを検出する方法およびシステムに関し、特に該パターンをデータ構造を考慮しながら、高速に自動的に検出する方法およびシステムに関する発明である。

【０００２】

【従来の技術】ワールド・ワイド・ウェブ（WWW）上の複雑にリンクが張り巡らされたデータや、自然言語で記述された文書の集まり（テキスト・データベース）のように、構造は持っているのだが、はっきりと固定された構造をもたないデータが最近増加している。このようなはっきりと固定された構造をもたないデータは半構造データと呼ばれる。半構造データでは、その構造自体が未知であるため、構造のパターンを検出するための固定した視点を与えることができない。一般に、複雑な構造を持つデータは、頂点と辺からなるグラフを用いて表現することが可能である。例えば自然言語で書かれた文は、構文解析と意味解析によって概念間のネットワークとして表現できる。また、会社の組織や社員は、関係しているエンティティ同士を結び合わせたグラフとして表現される。このようなグラフの集合がデータベースとして与えられたとき、そこからパターンとして頻繁に出現する部分グラフを検出できれば、データ集合に含まれた重要な概念を抽出することができる。しかし、この問題を高速に解くアルゴリズムは知られていない。

【０００３】データ集合からデータ構造のパターンを検出する方法としてデータマイニングが知られている。このデータマイニングで検出できるパターンとしては、相関ルール(Association Rules) が良く知られている。従来、相関ルールを検出する際には、データベースをトランザクションの集合として捕らえ、トランザクションをアイテムの集合と考える。そして、データベース中のトランザクションに頻繁に現れる（共起する）アイテムの集合を求め、その頻出アイテム集合から相関ルールを

導き出す。この場合、対象データは単純なアイテムの集合であり、導き出されるパターンも単純なアイテムの集合である。また複雑なデータ構造に対しては、ユーザが視点を与えることにより、構造は単純化することができる。従来、複雑なデータに対するデータマイニングの適用は、このような固定された視点を与えることにより単純化したデータに対して行われてきた。しかし、これは固定された視点を与えることができる場合に限られていて、複雑な構造の中のどこに注目してよいかあらかじめ分からないような状況では、このような単純化は容易ではない。逆にいえばデータマイニングにおいては、その目的が未知の知識の検出やデータの分析にあるため、あらかじめ固定された視点を与えることが難しいという問題がある。また固定された視点に注目してパターン検出を行うには、予めデータ構造を分解してフラットな形式のデータに変換するので、データ構造を有効に用いたパターン検出がなされない。また注目されなかった部分は当然検出対象から外れてしまう。

【０００４】

【発明が解決しようとする課題】従って、本発明が解決しようとする課題は、データの集合からその中に含まれる重要なパターンを検出する方法及びシステムを提供することである。また別の課題は、データ集合からその中に含まれる重要なパターンを、高速に短時間で検出する方法及びシステムを提供することである。また別の課題は、データ集合からその中に含まれる重要なパターンを、そのデータの構造を有効に用いて検出する方法及びシステムを提供することである。また別の課題は、固定された視点を与えることなく、データ集合をマイニングする方法およびシステムを提供することである。

【０００５】

【課題を解決するための手段】本発明は上記課題を解決するために、木構造データで表わされたデータ集合を含むデータベースから、集計対象となる候補パターンを用いて、頻出パターンを検出するシステムであって、

（１）データベースから候補パターンにマッチするパターンを集計する手段と、（２）前記集計により出現頻度の高いパターンを検出する手段と、（３）前記検出したパターンから、次の集計対象となる候補パターンを生成する手段と、を有する。図１に本発明の概要を示す。木構造データで表わされたデータ集合を含むデータベースから、頻出パターンを検出するにあたり、ブロック１１０でデータベースから候補パターンにマッチするパターンを集計し、ブロック１２０では、集計により出現頻度の高いパターンを検出し、ブロック１３０では、検出した出現頻度の高いパターンから、次の集計対象となる候補パターンを生成するように構成する。また別の態様として、本発明は大量のテキストデータから、有用な概念を抽出する、テキストマイニング・システムであって、

（１）各テキストデータ中の文章を構文解析する手段と、（２）前記構文解析の結果に基づき構文木を生成する手段と、（３）前記構文木の集合からなる、データベースを作成する手段と、（４）データベースから集計対象となる候補パターンを集計する手段と、（５）前記集計により出現頻度の高いパターンを検出する手段と、（６）前記検出したパターンから、次の集計対象となる候補パターンを生成する手段、を有する。

【０００６】

【発明の実施の形態】本発明の実施の形態を説明する前に、若干の言葉の定義を行う。

［準備］$I = \{ i_1, i_2, \ldots, i_n \}$ をアイテムと呼ばれるリテラルの集合とする。ラベル付き集合 $g = (l, A)$ とは、ラベル $l \in I$ とラベル付き集合の集合 $A = \{g\}$のペアになっている。また、ラベル付き集合をグループと呼ぶ。 $G = I \times set(G)$ はグループのドメインを示す。トランザクションの集合を D とする。各トランザクション T は、グループの集合（$T \subseteq G$ ）である。各トランザクションおよびトランザクション中の各グループは固有の識別子を持つ。"構造化された相関パターン" (structured association pattern)、あるいは、簡単に"パターン"もグループの集合である。なお、明細書中において、ラベル l だけからなるグループ $g=(l,\{\})$を単にアイテム l と呼ぶ場合もある。

【０００７】次にパターンマッチングについて説明する。グループ $g = (l_g, A_g)$ が他のグループ $h = (l_h, A_h)$ にマッチするとは、両者のラベルが等しく（$l_g = l_h$ ）、かつ、 A_g が A_h の部分集合である（$A_g \subseteq A_h$ ）ことをいう。パターン $P = \{ g_1, g_2, \ldots, g_n \}$ がトランザクション $T = \{h_1, h_2, \ldots, h_n\}$ にマッチするとは、パターン P 中の各要素 $g_i \in P$ に対して、それぞれマッチするグループ $h_i \in T$ が別々に存在していることをいう。ここで、複数のグループ同士のマッチングでは、それぞれ別々のグループ同士がマッチしなければならないことに注意する。例えば、図２に示すようなパターン $P_1 = \{(1, \{2\}), (1, \{4\})\}$ は、トランザクション $T = \{(1, \{2, 3, 4\}), (1, \{4, 5)\}\}$ にマッチするが、 $P_2 = \{(1, \{2\}), (1, \{3\})\}$ は T にマッチしない。なぜならば、P_2 の二つのグループは、どちらもトランザクション T の最初のグループにしかマッチしないからである。

【０００８】次にサポート（出現頻度）について説明する。パターン P が、トランザクションの集合 D に対してサポート s を持つとは、パターン P がトランザクション集合 D のうちの s% のトランザクションにマッチすることをいう。ただし、あるパターンがトランザクションに複数の方法でマッチしても、そのトランザクションに対するサポートのカウントは１でしかない。例えば、パターン $P = \{(1, \{2\}), (2,\{4\}), (2, \{5\})\}$ は、トランザクション $T = \{(1,\{2,3\}), (2,\{3,4\}), (2,\{3,5\}),(2,\{4,5\})\}$ にマッチするが、図３の A) の

ようなマッチングの仕方以外にも B) や C) のようなマッチングも存在する。このように複数の方法でマッチする場合もサポートのカウントは1でしかない。

【0009】次に頻出パターンについて説明する。構造化された相関パターンをマイニングするという問題は、トランザクションの集合 D が与えられた時、ユーザが与えた最小サポート値(minsup)よりも多くのトランザクションとマッチするような全ての構造化されたパターンを生成することである。もしパターンのサポートが minsup よりも大きければ、そのパターンを"頻出する" (frequent)と言い、そのパターンを頻出パターンと呼ぶ。X と Y をパターン P のサブセットとして、$X \cup Y = P$ かつ $X \cap Y = \phi$ の時に、パターン P から $X \Rightarrow Y$ という形式をしたルールを導出することができる。従って、本発明により構造化されたパターンを発見することにより、構造化されたルールの導出を行うことも可能である。各グループは集合として捉えられており、集合内のアイテムは、シーケンスについては考慮せず、一般性を失うことなく、ある決められた規則でソートされていると仮定する。なお本明細書中では、辞書順にソートされているものとして説明を行なう。同様に、各トランザクションやパターンのグループは、各グループのラベルの辞書順にソートされていると仮定する。もしグループ $g_1 = (l_1, A_1)$ とグループ $g_2 = (l_2, A_2)$ が同じラベルを持つ、即ち、$l_1 = l_2$ であるならば、アイテム集合 A_1 と A_2 を最初の要素から辞書順に比較していき、最初に違う要素が現れた時点で順位付けを行うことができる。トランザクション中に複数のグループが同じラベルを持つことを許しているので、上記のようなケースが出現する。また各グループに対して、null という特別なラベルを割り当てることによりラベルを持たないような単なる集合の集合を扱うことも可能である。

【0010】[パターン抽出システム]本発明の頻出構造化相関パターン（頻出パターン）の抽出システムについて説明する。本発明の頻出パターン抽出システムのブロック図を図4に示す。図4において、データベース(410)には、トランザクションの集合（データ集合）が格納されている。1件のトランザクションとは、1件の構造化されたデータ(木構造データ)のことを言う。頻出パターンを抽出する本システムへの入力は、大量のトランザクションが格納されたデータベース(410)である。集計装置(420)は、パターンの出現頻度を集計する装置であり、データベース(410)と候補パターン集合(450)を入力とし、頻出パターン集合(430)を出力する、集計装置(420)による集計の結果、候補パターン集合(450)の中で、出現頻度（サポート）の高いパターンのみを取り出して格納したものが、頻出パターン集合(430)である。候補生成装置(440)は、サイズ k の頻出パターン集合(430)を入力として、次の集計対象となるサイズ k+1 の候補パターン集合(450)を出力する装置である。候補パターン集合(450)には、集計装置(420)が集計の対象とするパターンが格納されている。ここには頻出パターンになりうる候補パターンが集められている。

【0011】本発明の頻出パターン抽出の流れを説明する前に、パターンのサイズを定義する。パターン中に含まれている（ラベルを含む）アイテムの合計をパターンのサイズとする。サイズ k のパターンを k-パターンと呼ぶ。サイズ k の頻出するパターンの全体の集合を L_k、サイズ k の候補パターンの全体の集合を C_k と記述する。図5に、本発明の頻出パターン抽出方法のフローチャートを示す。またこのフローチャートでは、パターンのサイズを変数 k に格納し管理する。

【0012】全体の頻出パターン抽出方法は以下の通りである。まず、はじめに単一アイテムからなるパターンの集計を行なう。また単一アイテムからなるパターン(サイズk=1 のパターン)の集計を行なうため、集計すべき候補パターン集合 C_1 の設定(510)を行ない、サイズを格納する変数k を1に設定する(ステップ520)。サイズ1の候補パターン集合 C_1 の設定(ステップ510)では、ラベルのみからなるパターンの集合 $\{l, \phi) \mid l \in I\}$ 及び、全てのラベルにマッチする特別なラベル "*" を持つ パターンの集合 $\{(*, \{i\}) \mid i \in I\}$を C_1 に入力する。ここで、ϕ は、空集合を表す。条件分岐(530)では、候補パターン集合の生成(ステップ510及びステップ550)で、生成された候補パターンC_k が空($C_k = \phi$)かどうかをチェックする。もし C_k が空ならば、ステップ570へ進む。 C_k が空でない場合、頻出パターンの集計作業が終わっていないので、ステップ540へ進む。 C_k が空でない場合、集計(ステップ540)を行なう。集計(ステップ540)は、候補パターン集合 C_k の要素である各パターンがマッチするトランザクションが何件あるかを求め、その中から出現頻度の高いものを頻出パターン集合 L_k として求める。集計(ステップ540)により頻出パターン集合 L_kが求められたら、次にサイズの大きさを1増やして、同様に集計を行なうため、サイズ k+1 の候補パターンの集合 C_{k+1} を生成する(ステップ550)。サイズを1増やして再び集計を行なうため、サイズを格納した変数 k を1増やし(ステップ560)、条件分岐(ステップ530)へいく。条件分岐(ステップ530)において、C_k が空になるまでサイズ k を1つづつ増やしながら、集計(ステップ540)及び候補パターン集合の生成(ステップ550)を繰り返す。もし条件分岐(ステップ530)で、C_kが空になった場合には(yesの場合)、ステップ570へ進む。ステップ570において最終的に求められた頻出パターン集合を出力する。最終的に求められた頻出パターン集合とは、サイズ1から k-1 の頻出パターン $L_1, L_2, \ldots, L_{k-1}$ の和集合である。(i が k-1 までなのは、C_k が空集合のため L_k も空集合だからである)

【0013】[候補生成]図4のブロック440及び図5のステップ550において、サイズ k の頻出パターン集合

L_k からサイズ k+1 の候補パターン集合 C_{k+1} の生成が行われる。以下に候補パターンの生成について詳細を述べる。図6に候補生成装置の詳細図を示す。候補生成装置は、あるサイズの頻出パターン集合から一つ大きいサイズの候補パターンの集合を生成する装置であり、入力として頻出パターン集合 L_k (610)を受け取り、候補パターン集合 C_{k+1} (650)を出力とする。頻出パターン集合610は、候補生成装置へ入力されるサイズ k の頻出パターン集合 L_k である。Join装置(620)は、サイズ k のパターン集合である L_k を入力として受け取り、Joinを行なって、サイズ k+1 のパターン集合を中間結果(630)として出力する。Joinの方法については、以降の段落で詳述する。中間結果630は、Join の結果、生成されたサイズ k+1 のパターンである。これは候補生成装置の中間結果である。Prune装置(640)は、Joinの結果として生成される中間結果(630)と頻出パターン集合 L_k (610)を入力として受け取り、Pruneを行なって、事前に頻出しないことが判断可能なパターンを篩い落してサイズ k+1 の候補パターン集合 C_{k+1} (650)を出力する。Prune の説明は、以降の段落で詳述する、候補パターン集合650は、Join 及び Prune の結果、生成されるサイズ k+1 の候補パターン集合 C_{k+1} であり、これが候補生成装置の出力である。なお候補パターン生成方法を説明する前に、若干の記載方法について説明する。

【００１４】［記載方法」もしパターン q がパターン p にマッチすれば、 q は p のサブパターンである。パターンとは木構造であり、パターン中の各グループはソートされているものとする。ここで、例として、パターンp={A{(a,{1,2}),(b,{3,4})}),(B,{c},),C} (図7参照) を考える。ラベル 1 だけからなるグループ(1,{}) は、 1 がリーフノードになっている。パターン中の要素1,2,3,4,c,C は、ラベルだけからなるグループのなので、リーフノードとなる。ここで、パターンの終端要素を次のように定義する。パターンのリーフノードで、かつ、兄弟ノードの中で、ソートしたとき最も後ろにくる要素を終端要素とする。ここで兄弟ノードとは親ノードが同一であるような子ノードの集合をいう(例えば、aとb)。このパターン p は、図7からもわかるように、4つの終端要素 {2,4,c,C} を持つ。

【００１５】候補パターンの生成は、Join フェーズと Prune フェーズの2つのフェーズからなる。本発明による方法について説明する前に、ここで、従来の方法によるJoinの方法について説明する。もしパターン a のある決められた位置のアイテムを除去して得られたサブパターンが、他のパターン b のある決められた位置のアイテムを除去して得られたサブパターンが同じならば、a を b とjoinすることで、候補パターンの生成を行う。よく知られた方法では決められた位置の要素として最後の要素を用いる。すなわち、パターン $a = \{a_1, a_2, \ldots, a_k\}$ とパターン $b = \{b_1, b_2, \ldots, b_k\}$ があると き、最後の要素だけが違う(即ち、$a_1 = b_1, \ldots, a_{k-1} = b_{k-1}$)場合に、a を b とjoinして、新しい候補パターン $c = \{a_1, \ldots, a_{k-1}, a_k, b_k\}$ を生成する。（但し $a_k < b_k$） この方法は、パターンの数が多くても、非常に効率良く機能する。しかしながら、構造化されたパターンに対しては、最後の要素あるいは、決められた位置の要素を除去することができない。なぜならば構造化されたパターンの制約に違反することになるからである。例えばパターン p = { (1, {3}), (1, {4}) } の場合を考える。joinしてパターン p を生成するようなパターンで、最後の要素だけが違うようなパターン a 、b は存在しない。そのようなパターンを生成するためには、$p_1 = \{(1,\{\}),(1,\{4\})\}$ と $p_2 = \{(1,\{3\}),(1,\{\})\} = \{(1,\{\}),(1,\{3\})\}$ をjoinする必要がある。しかしながら、 p_1 を p_2 とjoinすると、別のパターン p'= {(1,{}),(1,{3,4})} が生成される。従って構造化されたパターンの候補生成は従来の単純な候補生成方法を適用することができない。

【００１６】［候補パターン生成方法］候補パターンの生成は頻出パターン集合 L_k を入力として、サイズ (k+1) の候補パターンの集合 C_{k+1} を出力する。図8にそのフローチャートを示す。図8において候補パターン生成方法は、 Join フェーズ(ステップ810)と Prune フェーズ(ステップ820)の２つのフェーズからなる。

【００１７】＜Joinフェーズ(ステップ810)＞このフェーズでは、 L_k と L_k をjoinしてサイズ (k+1) のパターンを生成する。もし、パターン p_1 の終端要素の一つを落としたサブパターンがパターン p_2 の終端要素の一つを落としたサブパターンに一致するならば、パターン p_1 と p_2 をjoinする。 p_1 と p_2 をjoinして生成される候補パターンとは、パターン p_1 にパターン p_2 から落とした要素を元の場所に付け加えたパターンである。例えば、L_k に図9で示すパターン１とパターン２がそれぞれ存在していたとする。このときパターン１から要素 e1 を落としたサブパターンとパターン２から要素 e2 を落としたサブパターンは一致する。従ってこれらをjoinしてパターン３を生成し、これを候補パターンとする。ここで、 e1 、 e2 はどちらも終端要素である。このようにしてjoinによって生成された候補パターンは、中間結果 C'_{k+1} として保持する。

【００１８】＜Pruneフェーズ(ステップ820)＞Joinフェーズによって生成された C'_{k+1} には、サイズ k+1 の候補パターンが含まている。サイズ k+1 の候補パターンに対しては、一つの要素を取り除くことで幾つかのサイズ k のサブパターンが生成され得る。そのサブパターンの中で、一つでもそのサポート値(出現頻度)が最小サポート値(閾値)よりも小さい場合には、（即ち頻出パターン集合 L_k に含まれていない場合には）、その候補パターンを C'_{k+1} から除去する。これを C'_{k+1} の全てのパターンについて調べ、最終的に残ったパターンを候補

パターン集合 C_k として出力する。あるパターンから生成されたサブパターンが頻出パターンでなければ、そのパターンは頻出することは不可能であり、このように頻出することが不可能なパターンを事前に除去することによって、無駄な集計作業を減らすことができ、効率よく集計作業を行なうことができる。

【００１９】［頻出パターンの集計］図１０は集計装置(図４のブロック420)の詳細図である。集計装置は、パターンの出現頻度を集計する装置であり、データベース(1010)及び候補パターン集合 C_k (1020)を入力として受け取り、頻出パターン集合 L_k (1050)を出力する。データベース(1010)には、トランザクションの集合が格納されている。集合1020は、候補パターンの集合 C_k である。木構造適合検査装置(1030)は、トランザクションを1つづつ読み込み、そのトランザクションにマッチするパターンがあるかどうかを、 C_k 中の全てのパターンと比較して調べ、トランザクションにマッチするパターンを全てカウンタ(1040)に入力する。カウンタ(1040)は、候補パターン集合 C_k の全パターンの出現回数を数える。具体的には、木構造適合検査装置(1030)から入力されるパターン(トランザクションにマッチするパターン)について、各パターンのカウンタを1増やす。頻出パターン集合(1050)は、全てのトランザクションについて、上記集計作業を行なった結果、候補パターン集合 C_k 中の全てのパターンのうち、カウンタによって集計された出現回数が閾値よりも低いパターンを候補パターン集合 C_k より除去して生成される頻出パターン集合 L_k である。

【００２０】［頻出パターンの集計方法］頻出パターンの集計方法は、基本的にデータベースからトランザクション t を1つづつ読みこみ、候補パターン集合 C_k 中の全てのパターンについて、トランザクションにマッチするか調べ、マッチするならば、そのパターンのカウントに１を加える。これを全てのトランザクションについて、繰り返す。トランザクション t が、候補パターンにマッチするかどうかは、木構造同士のマッチングを行うことで判定する。図１１に頻出パターンの集計方法のフローチャートを示す。はじめに初期設定(ステップ1110)を行なう。これには、カウンタにおいて、 C_k 中の全てのパターンについて、そのカウントを0に設定することも含まれる。続いて、全てのトランザクションを読み込んだかチェックする(ステップ1120)。yes ならば、集計作業は終了で、ステップ1170へいく。noならば、ステップ1130へいく。ステップ1130においてマッチするパターンを一時的に格納する集合 Ct の初期化を行なう(Ct を φ にする。)。ステップ1140において、データベースからトランザクション t を1つ読み込む。ステップ1150において、 C_k 中の全てにパターン p について、読み込んだトランザクション t にマッチするかどうか調べる。もしマッチするならば、そのパターンを C_k へ加える。ステップ1160において、 Ct に含まれる全てのパターンについて、これらのカウントを1増やす。ステップ1160が終了したら、ステップ1120へ行く。これを全てのトランザクションについて繰り返す。全てのトランザクションについて終了したら、ステップ1170 へ進む。ステップ1170において候補パターン集合 C_k の全てのパターンについて、そのカウントが閾値(最小サポート値)以上であるかどうか調べ、閾値以上のパターンについては、 L_k に加える。結果として、求められた頻出パターン集合 L_k が出力される。

【００２１】［２段階の木構造パターンの抽出］一般の木構造データに対する頻出パターンの抽出は、上記の方法で実現することができる。しかし、木構造に対して、図１２に示すような２段階の木構造に限定すると、前述の一般の木構造よりもさらに効率よく高速に計算することが可能である。まずグループラベル及びアイテムの用語について説明する。２段階の木構造データでは、パターンはラベル付き集合の集合であり、ラベル付き集合 g =(l,A) とは、ラベル $l \in I$ とアイテム集合 $A \subseteq I$ のペアである。グループラベルとは、ラベル付き集合のラベル l を指し、アイテムとは、ラベル付き集合のアイテム集合の要素を指す。図１２において、A,B,C がグループラベルであり、1,2,...,8 はアイテムである。またラベル L と空のアイテム集合{} からなるグループ(L,{})において、L がグループラベルとなる。

【００２２】［２段階の木構造パターンの抽出システム]２段階の木構造の場合の頻出パターン抽出システムのブロック図及びフローチャートは、一般の木構造の場合と同様で、図４及び図５に示されているものと同様である。ただし、効率よく処理を行なうため、候補パターン生成装置(図４ブロック440)及びその実装方法(図５ステップ550)、集計装置(図４ブロック420)及びその実装方法(図５ステップ540)が異なっている。本集計装置はパターン中の複数のグループが同じグループラベルをもつ場合も処理することが可能になっている。２段階の木構造データに対する候補生成方法、効率的な集計方法について以降の段落で記述する。

［２段階の木構造に対する候補パターン生成方法］

【００２３】２段階の木構造の場合のパターンの抽出の概略は、任意段数の木構造と同様で、図８および図６に示されている。ただし効率よく計算を行なうため、Join の方法が一部異なる。

＜２段階の木構造に対するJoin方法＞サイズ (k-1) の二つのパターンをjoinをしたとき、どんなサイズ k のパターンが生成されるのかを考える代わりに、サイズ k のパターンは、どうようなサイズ (k-1) の二つのサブパターンに分割できるかについて考える。サイズ $k \geq 3$のパターンを、最後の2つの要素に基づいて、次の4つのタイプに分類する。それぞれのタイプにおいて、サイズ (k-1) の2つのサブパターンをjoinしてサイズk のパ

ターンが生成できるように、サイズ k のパターンをサイズ (k-1) の2つのサブパターンに分割する。

・Type A: パターンの最後の2つの要素が、同じグループに属するアイテムである。

・Type B: パターンの最後の要素がグループラベルで、最後のグループはグループラベルだけからなる。かつ、最後から二つ目の要素が、最後から2番目のグループの要素である。

・Type C: パターン最後の要素が最後のグループのアイテムで、最後から2番目の要素がグループラベルである(最後のグループは、このグループラベルとアイテムからなる)。さらに、このタイプは、最後から3番目の要素に基づき、Type C1 とType C2 に分類される。

・Type D: 最後の二つの要素がグループラベルである(すなわち、最後の2つのグループは、どちらも、グループラベルだけのグループである)。

【００２４】図１３は、これらの分類とサイズ k のパターンを生成するためにjoinするサイズ k-1 のサブパターンを示している。この図において、三角形はサブパターンおよびサブグループを示している。各要素はグループラベルかアイテムのいずれかなので、全てのパターンは、上記の4つパターンにどれかに、必ず分類される。従って、どのようなパターンを生成する場合であっても、図１３のいずれかの式を使ってパターンを生成することができる。最後と最後から２番目のグループが同じグループラベルを持つような場合には、グループの順序が変わるため注意する必要がある。候補生成は、任意段数の木構造の候補生成と同様に次の２つのフェーズからなる。

【００２５】＜Joinフェーズ(図８ステップ810)＞頻出パターン集合 L_k を L_k とjoin してサイズ k+1 のパターン集合を生成する。もしパターン $p_1 \in L_k$ が、図１３の右側のサブパターンのどれかと一致するならば、パターン $p_2 \in L_k$ を探し、p_1 と p_2 をjoinして、新しい候補パターンを生成して、C'_{k+1} に入れていく。L_k を事前に辞書順にソートしておくことで、p_1 のjoinの相手は、その範囲が限定され、p_1 の相手を探すために、L_k全部を探す必要がない。

【００２６】＜Pruneフェーズ(図８ステップ820)＞サイズ k+1 の候補パターンから生成され得るサイズ k のサブパターンのサポート値が最小サポート値より小さい場合には、この候補パターンを C'_{k+1} から除去する。効率良く全ての k-サブパターンが L_k の要素であるかどうかを効率的に見るために、L_k を格納するハッシュ木を用いる。図１４は、join とpruneが終わった後の L_3 と C_4 の例を示している。joinフェーズにおいて、パターンp_1 は、 p_3 とjoin して c_1 と c_2 を生成する。パターン p_3 は、 p_5 とjoinして c_3 が生成される。また、 p_5 は、 p_6 とjoinして c_4 が生成される。pruneフェーズにおいて、 c_1 のサブパターン {(1,{3,5})} と c_2 のサブパターン {(1,{}), (1,{}), (2,{})} が L_3 にないので c_1 と c_2 は除去される。

【００２７】［２段階木構造データの集計］２段階の木構造データに対する集計装置やその方法は、任意段数の木構造データに対する集計装置や方法と若干異なる。パターンのサポートを数え上げるため、パターンがトランザクションにマッチしているか否かを効率よく調べる必要がある。トランザクションおよびパターンは、後述する方法で、シーケンスに変換され、パターンのシーケンスが、トランザクションのシーケンスのサブシーケンスになっているかどうかを調べる(図１５参照)。パターン及びトランザクションから変換したシーケンス中のアイテム I がマッチングに使われる場合には、それぞれそのアイテム I を含むグループのグループラベル L もマッチングに使われなければならないことに注意する。基本的には、上記の方法でマッチングを調べるが、同じグループラベルを持つグループを許しているため、図１６の例のように、サブシーケンスでない場合にもマッチすることが起こりうるため、同じグループラベルを持つグループに対しては、特別な処理を行う必要がある（詳細は以降のクラスタのマッチングの説明段落で記述する）。

［２段階木構造データの集計装置］２段階の木構造データに対する集計装置のブロック図を図１７に示す。２段階の木構造データに対する集計装置は、入力としてデータベース(1710)、候補パターン集合 C_k (1720) を受け取り、それらのパターンにマッチするトランザクションが何件あったかを集計し、閾値以上出現したパターンを頻出パターン集合 L_k (1790)として出力する。データベース1710は、入力となるデータベースでありトランザクションの集合が格納されている。集合1720は、候補パターン集合 C_k である。データベース(1710)及び候補パターン集合(1720)が集計装置への入力となる。ブロック1730は、トランザクション用シーケンス生成装置である。データベース(1710)から読み込んだトランザクションは、トランザクション用シーケンス生成装置(1730)によって、トランザクション用シーケンス(1735)へと変換される。その変換方法については、後述する。シーケンス1735は、トランザクション用シーケンスであり、トランザクション用シーケンス生成装置(1730)により生成されたものである。また、トランザクション用シーケンス(1735)は、シーケンス適合検査装置(1740)へ入力される。ブロック1740は、シーケンス適合検査装置で、トランザクション用シーケンス(1735)を入力とし、読み込んだトランザクションにマッチするパターンがあるかどうかを候補パターン用ハッシュツリー(1780)を用いて調べる。シーケンスのマッチングを検査した結果、トランザクションにマッチするパターンがあれば、そのパターンをカウンタに渡す。ブロック1750は、候補パターン集合 C_k 中のパターンが何個のトランザクション中に現れていたの

かを集計するカウンタである。カウンタ(1750)は、シーケンス適合検査装置(1740)から入力されたパターンのカウントを1増やす。ブロック1760は、候補パターン用シーケンス生成装置であり、候補パターン集合 C_k 中のパターンからシーケンスを生成する。シーケンス1765は、候補パターン用シーケンス生成装置(1760)で生成された候補パターン用シーケンスである。候補パターン用シーケンス(1765)は、ハッシュツリー生成装置(1770)へ入力される。ハッシュツリー生成装置(1770)は、候補パターン用シーケンス生成装置(1760)で生成された候補パターン集合C_k (1720)の全てのパターンに対する候補パターン用シーケンス(1765)を入力として受け取り、候補パターン用ハッシュツリー(1780)を生成する装置である。ハッシュツリー1780は、ハッシュツリー生成装置(1770)によって、生成された候補パターン用ハッシュツリーであり、シーケンス適合検査装置へ入力される。ブロック1790は、集計装置が、全てのトランザクションについて、マッチするパターンがあるかどうか、候補パターン集合 C_k (1720)の中のパターンについて調べ、マッチするパターンの出現回数について集計した後に、頻出した、即ち閾値(最小サポート値)以上出現したパターンとして出力する頻出パターン集合 L_k である。シーケンス適合検査装置(1750)については、以降でさらに詳しく説明する。

【００２８】［２段階木構造データの集計方法］２段階の木構造データに対する集計方法のフローチャートを図１８に示す。図１８は、任意段数の木構造データに対する集計のフローチャート(図１１)と非常に良く似ている。２段階の木構造データに対する集計は、次のように行なう。はじめに、候補パターン集合 C_k 中の全パターンについて、候補パターン用シーケンスの生成を行ない、これをハッシュツリーへ格納する。あとは基本的に全てのトランザクションについて一つづつ読み込み、トランザクション用シーケンスに変換しては、ハッシュツリーを用いてマッチするパターンを探し、マッチしたらカウントを1増やすという操作を繰り返し、最後に閾値(最小サポート値)以上のパターンについて、頻出パターン集合 L_k として求める。

【００２９】図１８において候補パターン集合 C_k 中の全パターンについて、候補パターン用シーケンスの生成を行なう(ステップ1810)。候補パターン用シーケンスの生成方法については後述する。次に、ステップ1810で候補パターン集合 C_k 中の全てのパターンについて生成された候補パターン用シーケンスをハッシュツリーへ格納する(ステップ1820)。ハッシュツリーへの格納方法については後述する。条件分岐(ステップ1830)で、全てのトランザクションが読み終えたかどうか調べ、yesならステップ1890へいく。no の場合には、全てのトランザクションについて調べ終わっていないので、ステップ1840へ進む。ステップ1840において、マッチするパターンを一時的に格納する集合 Ct の初期化を行なう(Ct を ϕ にする)。ステップ1850においてデータベースからトランザクション t を1つ読み込む。ステップ1860において、読み込んだトランザクション t からトランザクション用シーケンスを生成する。トランザクション用シーケンスの生成方法については後述する。ステップ1870において、ハッシュツリーを用いて、トランザクション t にマッチするパターンがあるかどうか調べる。もしマッチするならば、そのパターンを Ct へ加える。ステップ1880において、Ct に含まれる全てのパターンについて、これらのカウントを1増やす。ステップ1880が終了したら、ステップ1830へ行く。この処理を全てのトランザクションについて繰り返す。全てのトランザクションに対する処理が終了したら、(ステップ1890へすすむ。ステップ1890において、候補パターン集合 C_k の全てのパターンについて、そのカウントが閾値(最小サポート値)以上であるかどうか調べ、閾値以上のパターンについては、 L_k に加える。

【００３０】＜候補パターンのシーケンスへの変換(ステップ1810)＞与えられたトランザクション t にマッチする C_k 中の候補パターンを効率よく見つけるために、候補パターンをハッシュ木に格納する。候補パターンが同じグループラベルを持つ複数のグループを持つ場合を扱うため、各パターン p を次の手続きによりシーケンス s(p) に変換する。パターン p が与えられた時の候補パターン用シーケンス生成の手順を図１９に示す。ステップ1910において、 s(p) を空にする。ステップ1920において、候補パターン中のすべての要素についてを調べたか調べ、yes ならステップ1970へ、no ならばステップ1930へいく。ステップ1930において、パターン p 内の次の要素 i を取り出し、ステップ1940 へ進む。ここで、パターン p = {(1,{2,3}),(4,{}),(5,{6})} があった場合、取り出される要素とは、順に、1,2,3,4,5,6 となる。ステップ1940の条件分岐において、もし、要素 i がグループラベルならばステップ1950へ進む。グループラベルでなければ、ステップ1945へ進む。ステップ1945において、シーケンスs(p) に I(i) を加える。ステップ1950の条件分岐において、もし要素 i と同じグループラベルがパターン p 中にある場合には、ステップ1960へいく。他に存在しない場合にはステップ1955へいく。ステップ1955においてシーケンス s(p) に L(i) を加える。ステップ1960において、シーケンス s(p) に C(i) を加える。 C(i) は、i というグループラベルを持つすべてのグループを表している。これらのグループを クラスタと呼ぶ。ステップ1970において、クラスタ内の全ての要素をスキップし、違うグループラベルを持つ次のグループへ移動する。ステップ1945、1955、1960のいずれかでシーケンス s(p) に要素を加えたら、ステップ1920へ戻る。これをすべての要素が終わるまで繰り返す。ただし、同じグループラベルが存在する

場合には、それらのグループの要素はスキップされる(ステップ1970)。ステップ1920において、全ての要素について処理が終了したら、ステップ1980へいく。ステップ1980において、求めたシーケンス s(p) を出力する。上記のように、３種類の要素、C（クラスタ），L（ラベル），およびI（アイテム）がある。例えば、パターン{(1,{2,3})，(2,{1,3})，(2,{2,4})，(3,{2})} は、シーケンス < L(1)， I(2)， I(3)， C(2)， L(3)， I(2) >に変換される。ここで、パターン中の2番目と3番目のグループがクラスタを形成している。パターンがクラスタを持つとき、集計の際に、クラスタを処理するための処理が必要である。

【００３１】＜ハッシュ木への候補パターンの格納(ステップ1820)＞まず C_k 中の全てのパターンのシーケンスをハッシュ木に格納する。ハッシュ木の内部ノードはハッシュ表になっている。シーケンス s(p) をハッシュ木に付け加える時、ルートノードから始めてシーケンスをスキャンしながら木を下に向かって降りていく。深さ d にあるノードにおいて、 s(p) の d 番目の要素をハッシュ関数に適用しながら、どの枝を辿っていくべきかを決める。このハッシュ木は、全パターンに対する共通の接頭辞は、ルートノードからの経路で共有されており、ルートノードからリーフノードへのパスで、パターンのシーケンスを識別することができる。図２０に例を示す。例えば、パターン < L(1)， I(3)， I(4)> と <L(1)， I(3)， I(5)> は、図２０において、ルートからノード L(1)， I(3) と経由してそれぞれリーフ I(4)，I(5) へ辿り着く。シーケンス中で共通の L(1)， I(3) は経路が共有されている。

【００３２】＜トランザクションのシーケンスへの変換(ステップ1860)＞トランザクション t が与えられると、はじめに、次の手続きを用いて t をシーケンス s(t) に変換する。トランザクション用シーケンス生成の手順を図２１に示す。ステップ2110において、 s(t) を空にする。ステップ2120において、トランザクション中のすべての要素についてを調べたか調べ、yes ならステップ2190へ、no ならばステップ2130へいく。ステップ2130において、トランザクション t 内の次の要素 i を取り出し、2140 へ進む。ここで、トランザクション t = {(1,{2,3}),(4,{}),(5,{6})} があった場合、取り出される要素とは、順に、1,2,3,4,5,6 となる。ステップ2140の条件分岐において、もし、要素 i がグループラベルならば、ステップ2150へ進む。グループラベルでなければ、ステップ2145へ進む。ステップ2145において、シーケンス s(t) に I(i) を加える。ステップ2150の条件分岐において、もし、要素 i と同じグループラベルがトランザクション中にあるかどうか調べる。ある場合には、ステップ2160へいく。他に存在しない場合には、ステップ2180へいく。ステップ2160の条件分岐において、シーケンス s(t) に既に C(i) が加えられているかどうか調べる。もし、既に s(t) に C(i) が加えられている場合には、ステップ2180へいく。まだ、加えられていない場合には、ステップ2170へいく。ステップ2170で、シーケンス s(t) にC(i) を加える。ステップ2180で、シーケンス s(t) に L(i) を加える。トランザクション t 中にグループラベル i が複数存在している場合には、s(t) に L(i) を加えると同時に、 C(i) も s(t) に付け加えられている。 以上の処理を行なったらステップ2120へ戻る。ステップ2120の条件分岐において、トランザクション中の全ての要素について処理が終了していたら、ステップ2190へいく。ステップ2190において、求めたシーケンス s(t) を出力する。同じラベルを持つグループが存在する場合、そのラベルは、クラスタとして、そして個別の要素として２度処理されている。

【００３３】［シーケンス適合検査装置］シーケンス適合検査装置(図１７ブロック1740)は、トランザクションから生成したトランザクション用シーケンスと候補パターン用ハッシュツリーに格納された候補パターン用シーケンスとのマッチングを調べる。ハッシュツリーに対して、必要に応じてクラスタのマッチングを取り、そのために二部グラフのマッチングを計算する機能を用いている。シーケンス適合検査装置(ブロック1750)について、その詳細を図２２に示す。図２２において、シーケンス2210はトランザクション用シーケンス(図１７の1735)である。ハッシュツリー2220は、候補パターン用ハッシュツリー(図１７の1780)である。ブロック2230は、シーケンス適合検査装置本体であり、基本的には、この装置によりシーケンスの適合を検査する。しかし、トランザクション中にクラスタが存在する場合には、クラスタ適合検査装置(2240)によりクラスタの適合について検査する。クラスタ適合検査装置(2240)は、二部グラフ適合検査装置(2250)を呼び出し、二部グラフによる適合検査を行なう。

【００３４】＜集計方法＞トランザクション t にマッチする全ての候補パターンを検出するために、次の手続きを適用する。適用する手続きは、ハッシュ木をルートから辿って行くとき、現在どのノードにいるかに依存している。

・ルートノード： s(t) の各要素をハッシュし対応するバケット中のノードに、この手続きを再帰的に適用する。トランザクション t にマッチするどんなパターン p に対しても、 s(p) の最初の要素は s(t) に存在しなければならない。 s(t) 内の全ての要素をハッシュすることで、 s(t) 内に存在しない要素から始まるパターンを無視することができる。

・I または L のラベルを持つ内部ノード： s(t) 中の要素 e をハッシュして、このノードに辿りついたとする。 s(t) の e の次に来る要素をハッシュし、対応するバケット中のノードにこの手続きを再帰的に適用す

る。

・C のラベルを持つ内部ノード： 二部グラフを構築し、トランザクション中のクラスタにマッチする候補パターンを検出する。 次に s(t) の要素 e をハッシュして、このノードにたどり着いたとする。 t にマッチするようなクラスタを持つ全てのパターン p に対して、この手続きを p の接尾辞の木に対して、再帰的に適用していく。

・リーフノード： リーフノードにたどり着けば、トランザクションにマッチするパターンを見つけたことになる。

【００３５】［クラスタのマッチング］ハッシュ木の C のラベルを持つノードに着目する。ノードは複数のクラスタを持ち、クラスタはパターンの一部分でその接頭辞が共通になっている。 $C = \{c_1, c_2, \ldots, c_k\}$ は、クラスタの集合を表す。このノードにたどり着いたとき、トランザクション t はクラスタ c(t) を含み、 c(t) は、 c_i (i=1..k) と同じグループラベルを持っている。目的は、c(t) にマッチする C に含まれる全てのクラスタを検出することである。単一のクラスタ c_i が、 c(t) にマッチするかどうかを決める方法について説明する。まず、c_i が n 個のグループを持っているとし、 c(t) が m 個のグループを持っているとする($n \leq m$)。 $c_i = \{g_1, g_2, \ldots, g_n\}$ および、 $c(t) = \{h_1, h_2, \ldots, h_m\}$ とする。 $g_x \in c_i$ が $h_y \in c(t)$ にマッチするならば、かつ、マッチする時のみ、エッジ $(g_x, h_y) \in E_i$ が存在するような二部グラフ $G_i = (c_i, c(t); E_i)$ を探す。図２３に二部グラフの例を示す。 c_i が c(t) にマッチするかどうかという判定は、G_iの二部マッチングの最大値のサイズが n であるかどうかをチェックすれば良い。図２３において直線は二部マッチの最大エッジを示している。マッチングにより c_i の全てのグループがカバーされていれば、この場合、c_i は c(t) とマッチする。効率よく、全ての $c_i \in C$ に対して、全てのグラフ $G_i = (c_i, c(t); E_i)$ を構築するため、 c_i ($i = 1, \ldots, k$)のすべての候補グループを格納するハッシュ木を事前に、構築しておく。各グループ $h_i \in c(t)$ に対して、ハッシュ木を使って h_i にマッチする全てのグループを識別しておき、対応するエッジをグラフに加えておく。上記の手続きをしてしまえば、全てのグラフがわかっており、トランザクション t にマッチする全てのクラスタを検出することができる。いくつかの場合において、二部グラフ G_i を完全に構築する前に、 c_i が c(t) とマッチするかどうかを見ることができる。

【００３６】［本発明を応用した実施例］

<テキストマイニング・システム>企業では、顧客の満足度の向上を図るため、コールセンターやお客様相談室と呼ばれる部門を設けて、電話等による顧客からの問い合わせやクレームに対して、質問に答えたり、正しい操作方法を教えたり、商品の改善などに役立てている。顧客から寄せられた内容は、電話応対したオペレータにより文章入力され記録され、蓄積されていく。蓄積された大量の記録文書を分析することにより、頻繁に問い合わせのある質問は、事前に質問と回答を用意し、オペレータがそれらの知識を共有することで、オペレータの負担や応答時間を減らし、コストの削減を図ることができる。また、商品の不具合などは、開発部門へどのようなクレームが多いのかを知らせることで、機能の改善などに用いることができる。顧客の要望を分析することで、新製品の開発に活かすことも可能である。顧客からの問い合わせ件数は、企業規模や商品の種類等に依存するが、多い企業では、月に数万件もの問い合わせが寄せられており、そこに蓄積される記録文書は膨大な量となっている。顧客からの問い合わせを分析するには、記録文書の全てに目を通して、どのような案件に対する問い合わせやクレームが多いのかを調べる必要がある。月数万件の問い合わせがある場合には、人手による全文書の分析は、ほとんど不可能であり、そのような場合には、全文書の中からサンプリングして一部の文書を取り出して調べるか、オペレータの主観的な判断を任せざるを得ない。

【００３７】通常、コールセンターのオペレータは、顧客からの電話の内容をデータベースに自然言語で打ち込む。上述のように、蓄積されたデータベースには顧客が頻繁に発する質問、要望、不満、賛辞などが記録されており、今後の企業の営業方針を決める重要な情報となる可能性がある。このような場合に本発明を用いたパターン検出システムを適用することにより、巨大なデータベースから重要なメッセージを抽出（テキストマイニング）することができる。文や文書を単語の集合と見立てて、単純に通常の相関ルールや時系列パターンの検出方法を適用することができるが、この方法は上手く行かない。なぜなら、構造を持たないパターンからもとの文の意味を想像することは難しい上に、大量のパターンが検出されてしまい、その中から重要なパターンを検出することは不可能に近いからである。一方、自然言語処理の技術により、単語間の関係を自動的に抽出することが可能になっている。機械翻訳の典型的な方法は、元の文を意味解析して、単語間の意味関係に基づいて構文木(ネットワーク)を構築し、その構文木上で変換を施した後に、別の自然言語の文に変換することにより、翻訳を行う。単語の意味はこの構造中の位置により決定されるので、このようなネットワークから、重要なパターンを検出する。図２４はそのようなネットワークの例を示す。このようなネットワークが大量に与えられたとき、その部分グラフをパターンとして検出する。

【００３８】まず、述語は文の意味を決定する上で最も重要な役割をになっているため、文中の述語に注目する。述語は、それと直接または間接に修飾関係にあるほ

かの単語を引数語として持つ。このような単語のグループを、「述語引数語組 (predicate-argument tuple)」と呼ぶ。例えば、図２４の文は、図２５に示す３つのグループを持つ。このようなグループは、元の文の全ての情報を保持してはいないが、単語の羅列と比較すると情報量が格段に違う。また元の文の意味を容易に想像することができる。このような述語引数語組は、ラベル付き集合で表現することができる。ラベルは述語で、集合の要素が引数語である。文や文書は、ラベル付き集合の集合と見なせる。したがって、このような方法で２段の木構造への変換を行うと、本発明で示した２段の木構造からの頻出パターン検出方法により高速に頻出パターンを探すことができる。例えば，{(install {software A, driverB})), (get {error}) }というようなラベル付き集合からなるパターンを、データベースから検出することができる。

【００３９】＜ＦＡＱ作成システム＞現在、顧客の声は、電話だけに限らず、電子メールやＷＥＢなどの様々なメディアを通して、企業に寄せられており、その数は非常に多く、これらの分析の自動化が望まれる。そこで文章入力された顧客からの情報を自然言語解析を用いて構造化されたデータを生成し、本発明によって頻出する構造化パターンを検出し、その結果を用いて、ＦＡＱ(Frequently Asked Question)の作成を支援するシステムを構築する。

【００４０】・データ生成

頻出する構造化パターンの抽出は、構造化されたデータから行なうので、文書データから構造化データを生成する必要がある。そのためには、まず、文書を構文解析し、構文木の構築を行なう。次に、構築された構文木データから構造化データを生成する。

【００４１】・構文木データの生成

構造化パターン抽出のためのデータは、全ての文書データに対して次の処理を行なうことにより、生成することができる。 以下に形態素解析、文節生成、係り受け構造の構築(構文木の構築)の処理を説明する。例えば、「私は会社に行く。」という文章に対して、形態素解析を行なうことで、以下の単語を切り出すことができる。

["私",名詞]
["は",助詞]
["会社",名詞]
["に",助詞]
["行く",動詞]

さらに文節生成を行なうと、次の文節が生成される。

["私","は"]
["会社","に"]
["行く",]

さらに、係り受け構造を抽出することで、以下の係り受け構造が構築される。このような係り受け構造を木の形で表現したものが構文木となる。

["私" -> "行く"]
["会社" -> "行く"]

以上のように、文章から構文木を構築することができる。これを全ての文章に対して行なう。

【００４２】・構造化データの生成

次に、構築された構文木から構造化データの生成を行なう。ここでは、例として、「デスクトップの画面までは立ち上がるが、プログラムを開こうとすると、前述の画面が表示される」という文章から構築された構文木を用いる(図２６を参照)。なお、この図において、濃色ノードは、動詞であることを示している。構造化データとは、複数のグループ（ラベル付き集合）からなる。グループ（ラベル付き集合）とは、ラベルになる要素及び0個以上の他の要素からなっており、FAQの作成支援ツールを構築するために、ここでは、動詞をラベルとして、その動詞に係りうけの関係を持つような他の単語をメンバーとするようなグループを構築する。また、そのようなグループを動詞の数だけ生成し、それを1件のデータとする。上記に構築した構文木において、ノード["デスクトップ"]、["画面"]は、動詞である["立ち上がる"]というノードの子孫のノードである。そこで、この3つのノードから動詞の["立ち上がる"]をラベルとして、他のノードをその要素とするようなグループ(ラベル付き集合) "立ち上がる" ("デスクトップ" "画面") を生成する。同様にして、["プログラム"]及び動詞ノード["開く"]からグループ "開く" ("プログラム") が生成される。ただし、ここで、["開く"]の子孫には、["デスクトップ"]や["画面"]などがあるが、そのノードへと辿る途中に、動詞ノードである["立ち上がる"]があるため、["開く"]と同じグループには含めないものとする。同様に、動詞ノード["表示する"]及び["画面"]["前述"]からグループ "表示する" ("画面" "前述") を生成する。このように、一つの文章から生成されたグループをまとめて、

【００４３】"立ち上がる" ("デスクトップ" "画面") "開く" ("プログラム") "表示する"("画面" "前述")

【００４４】という構造化データが構築される(図２７を参照)。このようにして、全ての文章から構造化データを構築する。また、後で、元の文書を読む必要があることから、構築された構造化データの識別子から元の文書が取り出せる様に、両者の対応関係を保持しておく。

【００４５】・頻出パターンの検出

上記で構築した大量の構造化データに対して、最小サポート値を閾値として与えて、本発明を用いて頻出構造化パターンを検出する。

・ＦＡＱ生成

FAQを生成するには、本発明によって検出された頻出パターンとそのパターンにマッチする構造化データを取り出し、さらに、その構造化データの元の文章を分析する必要がある。まず検出された頻度の高い構造化パター

とその頻度の一覧を表示する。表示された複数のパターンの中から任意のパターンを選択して、各文書から作成された構造化パターンのうち、そのパターンにマッチするデータだけを抽出し表示する。例えば次のパターンを選択する。

"起動する" ("再")　　"切る"("電源")

すると、上記パターンにマッチする構造化データが複数表示される。表示された構造化データの中から1つのデータを選択すると文書ブラウザ(図２８を参照)が起動され、そのデータの元の文章を見ることができる。選択されたデータの元文書である「電源を切って再起動したらスキャンディスクで問題が見つかった」という文書が、図２８に示す文書ブラウザの上部に表示される。またブラウザの下部に、その文書がどのように形態素解析され、係り受け構造の構築がされたのかがわかるように、その係り受け構造を視覚的に表示する。実際に、FAQの作成支援を行なうためには、構文解析や個々のデータを見るよりも頻出するパターンにマッチする元文書の一覧を、直接見ることの方が有益であり、元文書の一覧を見る機能を提供する。さらに頻出パターンの元文章だけでなく、元文章の前後の文章を見る機能を提供する。つまり元文章を含む文書全体を見る機能を提供する。

【００４６】<テキスト分類システム>大量の文書を扱う際、文書を自動的に分類したいという要求は極めて大きい。本発明の頻出パターン検出方法を用いて、テキストを分類する事ができる。構造化されたパターンは可読性が高いので、結果として得られるテキストの分類がどういう物であるかを説明することが容易である。テキストの自動分類例として、電子メールの自動的な仕分けシステムを構築できる。1日に何十通もの電子メールが到着するような業務を行う場合、メールのサブジェクトに応じたメールフォルダに分類しなければ必要なときに必要なメールを探し出すことが難しくなる。この分類作業を人手で行うことは、メールが多くなると困難になる。電子メールを扱うソフトウェアは通常、ユーザがフォルダ（カテゴリ）に電子メールを分類して保存する機能を提供している。到着した電子メールは通常「受信」（Inbox）と呼ばれる特別なフォルダに入れられ、ユーザがメールの内容にしたがって、関連するフォルダ（カテゴリ）に移動し保存する。到着するメールの数が少ないうちはあまり問題にならないが、1日に何十通も到着するような場合、この分類の手間は大きくなる。そこで、本発明を応用して自動分類器を構成する。この電子メール自動分類の処理の流れを以下に説明する。

【００４７】（ａ）ユーザが現在手持ちのメールの一部を手動でフォルダ（カテゴリ）に分類する

（ｂ）ステップ（ａ）で手動分類したメールの集合を最初の分類例として、分類のルールを作成する。まず各メールを自然言語処理により述語と引数語組の集合に分解し、その結果t_iを得る。各メールの分類先フォルダを f とすると、t_i にf をラベルとする空の集合 f() を加える（ $t_i := t_i \cup \{f()\}$ ）。得られた t_i (i = 1, ..., n) に本発明を適用して、頻出パターンを検出する。最後に結論部にフォルダ（カテゴリ）名をラベルとする項を持つルールを生成する。これらのルールを r_1, ..., r_k とし、それぞれのルールの確信度（ルールが成立する確率）を計算しておく。

（ｃ）ユーザが手動で分類しなかった残りのメールを（ｂ）のルールに従い自動的に分類する。まずメールを自然言語処理により述語と引数語組の集合に分解し、その結果をｔとする。次に（ｂ）で求めた各ルール r_i について、がｔとマッチするかどうか調べ、マッチする場合、その程度を r_i の確信度にしたがってスコアを計算する。求めたスコアがもっとも高いフォルダ（カテゴリ）にメールを分類する。

（ｄ）自動分類の結果に誤りがあった場合、ユーザはその誤りを手動で修正する。手動で修正されたメールの分類を（ａ）で作ったフォルダに加えて、（ｂ）から繰り返すことにより、自動分類器の精度を段階的に高めていく。

【００４８】本発明に係るシステムは、その他、問い合わせや問題処理の担当者の自動選定を行うシステムにも応用できる。電子的なメディア(メール、Web)での問い合わせやトラブル処理の窓口では、さまざまな種類の問い合わせが大量に届けられる。一人の人間ではすべての種類の問い合わせに適切に回答できないので、その内容に応じた担当者に問い合わせを割り当てなければならない。この作業を自動化するシステムに応用できる。また新聞記事、本、論文など文書データベースの検索支援システムにも応用できる。大量の文書が電子化されているデータベース全体に対して文書検索を行うと、ユーザが注目しているものと関係ない文書が検出され、本来探し出したい文書がその中に埋もれてしまう（Web の検索エンジンのように）。データベース中の文書が予めいろいろな視点から分類されていれば、注目している分類中の文書に限った検索を行うことが可能となる。大量の文書を、かつ視点を変えて分類することが可能になる。その他、本発明の本質から離れることなく種々の応用例が考えられることは言うまでもない。

［ハードウェア構成例］図２９に本発明のシステム（頻出パターン検出、テキストマイニング、ＦＡＱ自動作成、テキスト分類）において使用されるハードウェア構成実施例を示す。システム１００は、中央処理装置（ＣＰＵ）１とメモリ４とを含んでいる。ＣＰＵ１とメモリ４は、バス２を介して、補助記憶装置としてのハードディスク装置１３（またはＣＤ－ＲＯＭ２６、ＤＶＤ３２等の記憶媒体駆動装置）とＩＤＥコントローラ２５を介して接続してある。同様にＣＰＵ１とメモリ４は、バス２を介して、補助記憶装置としてのハードディスク装置３０（またはＭＯ２８、ＣＤ－ＲＯＭ２９、ＤＶＤ３１

等の記憶媒体駆動装置）とＳＣＳＩコントローラ２７を介して接続してある。フロッピーディスク装置２０はフロッピーディスクコントローラ１９を介してバス２へ接続されている。

【００４９】フロッピーディスク装置２０には、フロッピーディスクが挿入され、このフロッピーディスク等やハードディスク装置１３（またはＣＤ－ＲＯＭ２６、ＤＶＤ３２等の記憶媒体）、ＲＯＭ１４には、オペレーティングシステムと協働してＣＰＵ等に命令を与え、本発明を実施するための頻出パターン検出プログラム、オペレーティングシステムのコード若しくはデータを記録することができ、メモリ４にロードされることによって実行される。これらコンピュータ・プログラムのコードは圧縮し、または、複数に分割して、複数の記録媒体に記録することもできる。

【００５０】システム１００は更に、ユーザ・インターフェース・ハードウェアを備え、入力をするためのポインティング・デバイス（マウス、ジョイスティック等）７またはキーボード６や、検出した頻出パターンや文書ブラウザなどを表示するためのディスプレイ１２を有することができる。また、パラレルポート１６を介してプリンタを接続することや、シリアルポート１５を介してモデムを接続することが可能である。このシステム１００は、シリアルポート１５およびモデムまたは通信アダプタ１８(イーサネットやトークンリング・カード)等を介してネットワークに接続し、他のコンピュータ、サーバ等と通信を行う。またシリアルポート１５若しくはパラレルポート１６に、遠隔送受信機器を接続して、赤外線若しくは電波によりデータの送受信を行ってもよい。

【００５１】スピーカ２３は、オーディオ・コントローラ２１によってＤ／Ａ（デジタル／アナログ変換）変換されたサウンド、音声信号を、アンプ２２を介して受領し、サウンド、音声として出力する。また、オーディオ・コントローラ２１は、マイクロフォン２４から受領した音声情報をＡ／Ｄ（アナログ／デジタル）変換し、システム外部の音声情報をシステムにとり込むことを可能にしている。ViaVoice（IBM商標）などのアプリケーションを用いて、頻出パターンの検出操作を音声コマンドによる操作で代用してもよい。

【００５２】このように、本発明のシステムは、通常のパーソナルコンピュータ（ＰＣ）やワークステーション、ノートブックＰＣ、パームトップＰＣ、ネットワークコンピュータ、コンピュータを内蔵したテレビ等の各種家電製品、通信機能を有するゲーム機、電話、ＦＡＸ、携帯電話、ＰＨＳ、電子手帳、等を含む通信機能有する通信端末、または、これらの組合せによって実施可能であることを容易に理解できるであろう。ただし、これらの構成要素は例示であり、その全ての構成要素が本発明の必須の構成要素となるわけではない。

【００５３】

【発明の効果】本発明により、複雑な構造を持つデータから、集合の集合、あるいは順列の集合のような複数段階の構造を持つパターンを自動的に全て検出することが可能になる。また検出されるパターンは従来の構造を無視したパターンより、元のデータの構造に近いので直感的に理解しやすい。ユーザは事前にパターンのテンプレートのようなものを与える必要がなく、さらに高速で、大量のデータに対しても短時間で処理が終了する。従って大量のデータベースから非常に重要なパターンが容易に検出できるようになる。

【図面の簡単な説明】

【図１】本発明の概要を示す図である。
【図２】パターンマッチングを説明する図である。
【図３】複数のマッチングを説明する図である。
【図４】本発明によるパターン抽出システムのブロック図である。
【図５】本発明によるパターン抽出のフローチャートである。
【図６】候補生成装置のブロック図である。
【図７】終端要素を説明する図である。
【図８】候補パターン生成フローチャートである。
【図９】Joinを説明する図である。
【図１０】集計装置のブロック図である。
【図１１】頻出パターン集計のフローチャートである。
【図１２】２段階の木構造の例である。
【図１３】パターンの分類を示す図である。
【図１４】候補パターンの生成例である。
【図１５】シーケンス・マッチングの１つの例である。
【図１６】シーケンス・マッチングのまた別の例である。
【図１７】カウンタのブロック図である。
【図１８】２段階の木構造の集計方法のフローチャートである。
【図１９】候補パターン用シーケンス生成手順である。
【図２０】ハッシュ木の例である。
【図２１】トランザクション用シーケンス生成手順である。
【図２２】シーケンス適合のブロック図である。
【図２３】二部グラフのマッチング例である。
【図２４】ネットワーク例である。
【図２５】述語引数語組の例である。
【図２６】構築された構文木の例である。
【図２７】生成された構造化データの例である。
【図２８】文書ブラウザの実施例である。
【図２９】本発明によるシステムで使用されるハードウェア実施例である。

【符号の説明】

１・・・ＣＰＵ
２・・・バス
４・・・メモリ

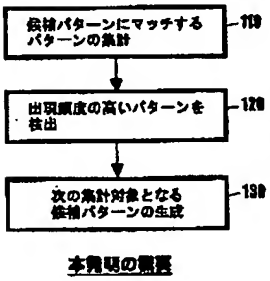
25

５・・・キーボード・マウス・コントローラ
６・・・キーボード
７・・・ポインティングデバイス
８・・・ディスプレイ・アダプタ・カード
９・・・ビデオメモリ
１０・・・ＤＡＣ／ＬＣＤＣ
１１・・・表示装置
１２・・・ＣＲＴディスプレイ
１３・・・ハードディスク装置
１４・・・ＲＯＭ
１５・・・シリアルポート
１６・・・パラレルポート
１７・・・タイマ
１８・・・通信アダプタ
１９・・・フロッピーディスクコントローラ

26

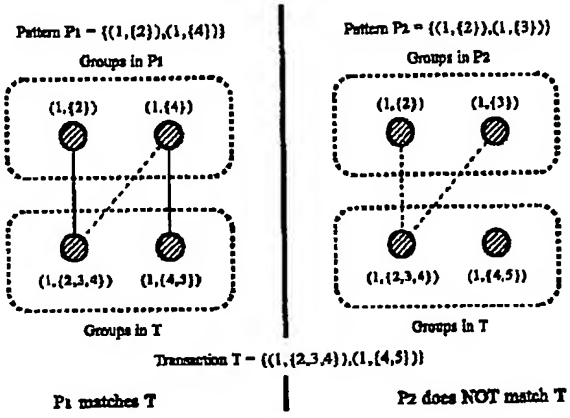
＊２０・・・フロッピーディスク装置
２１・・・オーディオ・コントローラ
２２・・・アンプ
２３・・・スピーカ
２４・・・マイクロフォン
２５・・・ＩＤＥコントローラ
２６・・・ＣＤ－ＲＯＭ
２７・・・ＳＣＳＩコントローラ
２８・・・ＭＯ
２９・・・ＣＤ－ＲＯＭ
３０・・・ハードディスク装置
３１・・・ＤＶＤ
３２・・・ＤＶＤ
１００・・・システム

【図１】



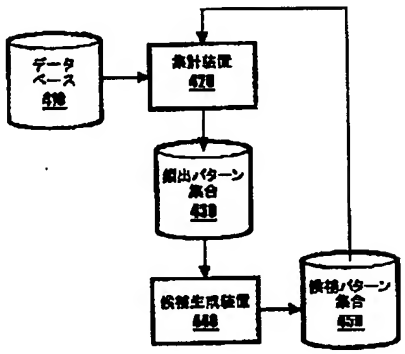
本発明の概要

【図２】



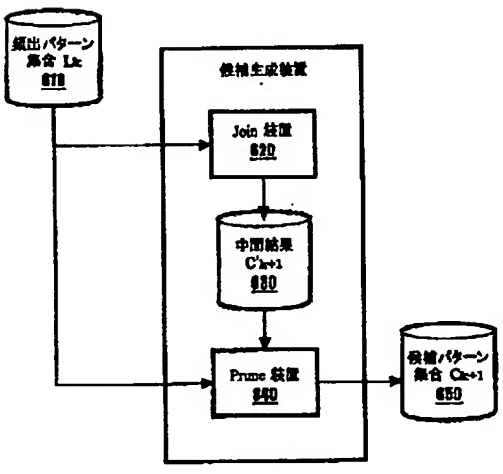
パターンマッチング

【図４】



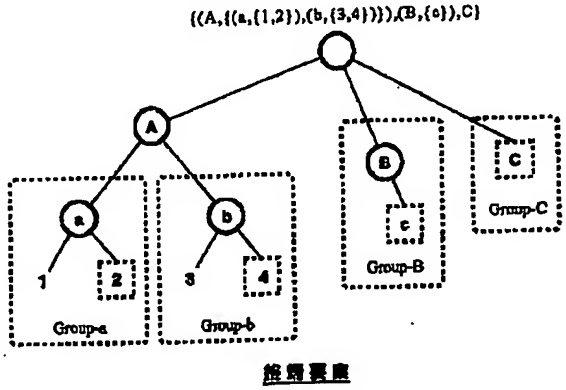
頻出パターン検出システムのブロック図

【図６】



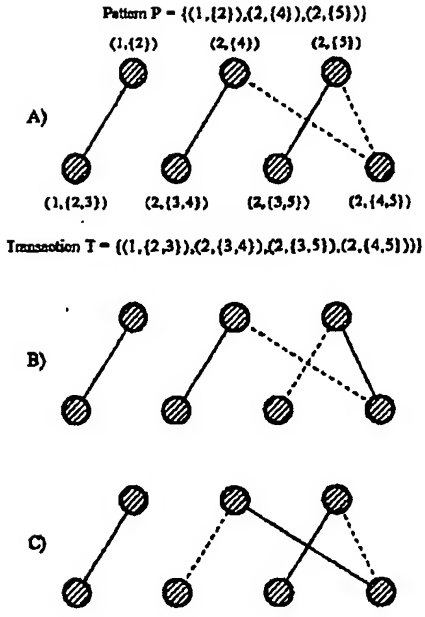
候補生成装置

【図７】



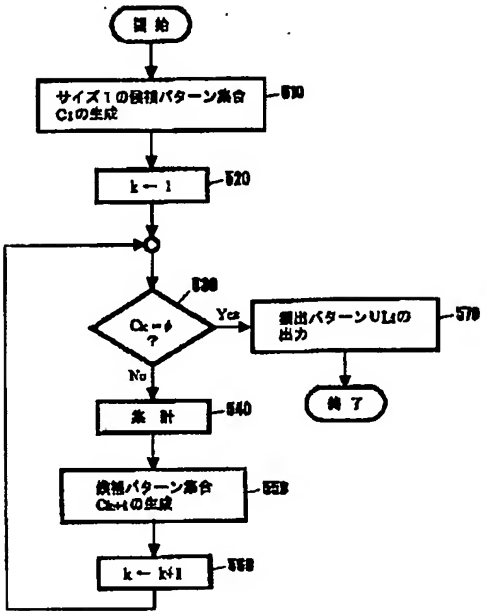
階層要素

【図３】



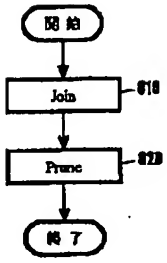
複数のマッチング

【図５】



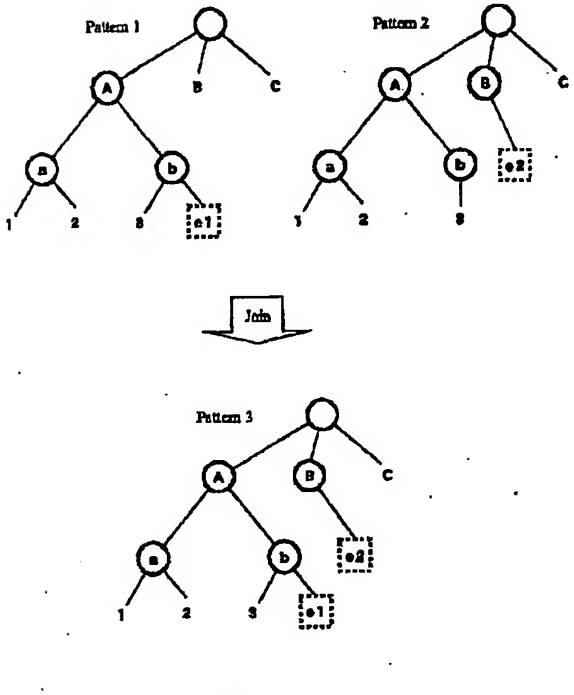
システム全体のフローチャート

【図８】

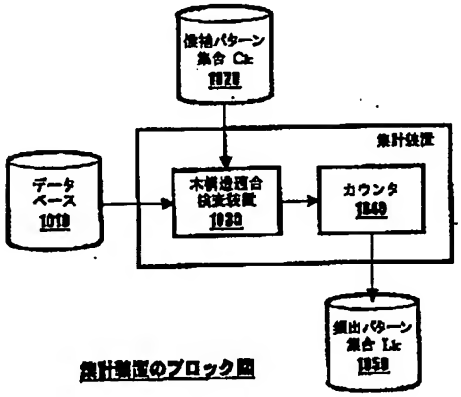


候補パターン生成フローチャート

【図９】

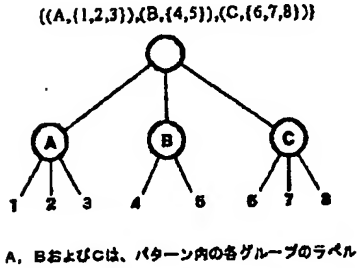


【図１０】



集計装置のブロック図

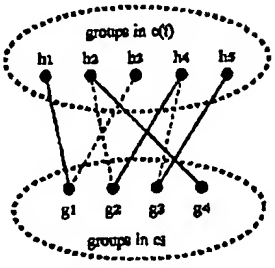
【図１２】



A，BおよびCは、パターン内の各グループのラベル

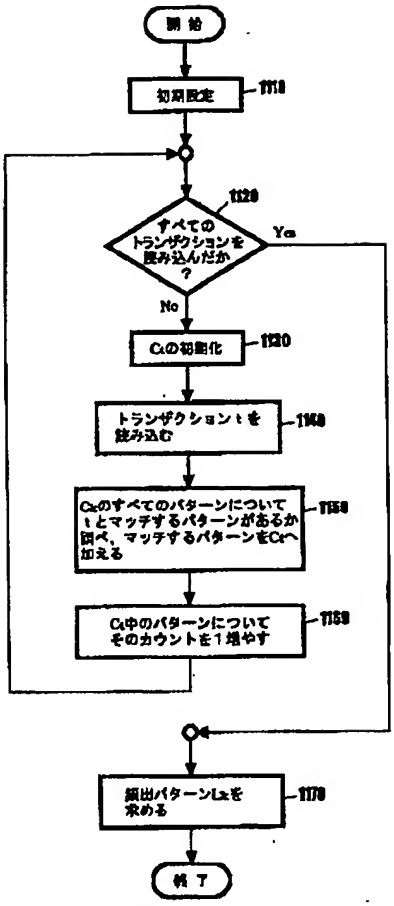
２段階に限定した木構造

【図２３】



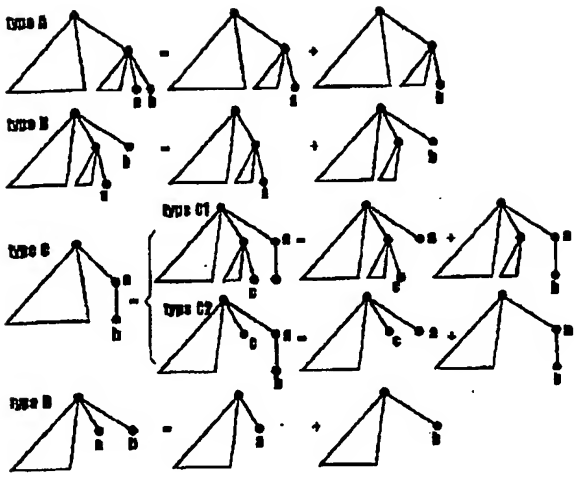
二部グラフのマッチング

【図１１】



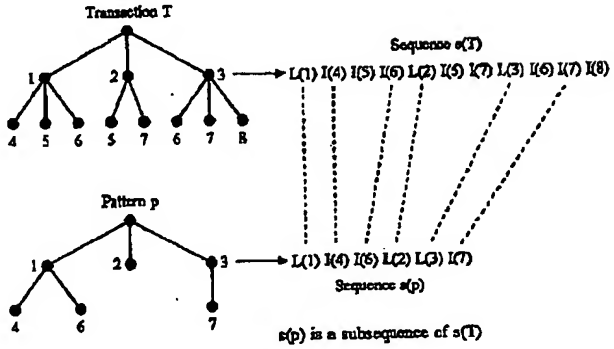
頻出パターンの集計

【図１３】



パターンの分割

【図１５】



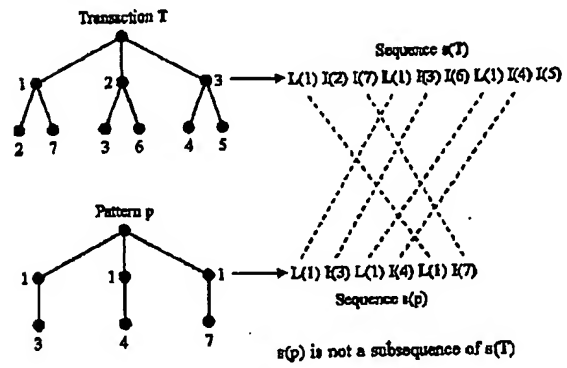
シーケンス・マッチング（１）

【図１４】

Frequent 3-patterns		Candidate 4-patterns			
		after join		after pruning	
p_1	{(1 {}) (1 {3})}	c_1	{(1 {}) (1 {3,5})}	c_3	{(1 {3}) (1 {5})}
p_2	{(1 {}) (1 {5})}	c_2	{(1 {}) (2 {}) (2 {4})}	c_4	{(1 {3}) (2 {4})}
p_3	{(1 {}) (2 {4})}	c_3	{(1 {3}) (1 {5})}		
p_4	{(1 {3}) (2 {})}	c_4	{(1 {3}) (2 {4})}		
p_5	{(2 {}) (2 {4})}				

候補パターンの生成例

【図１６】



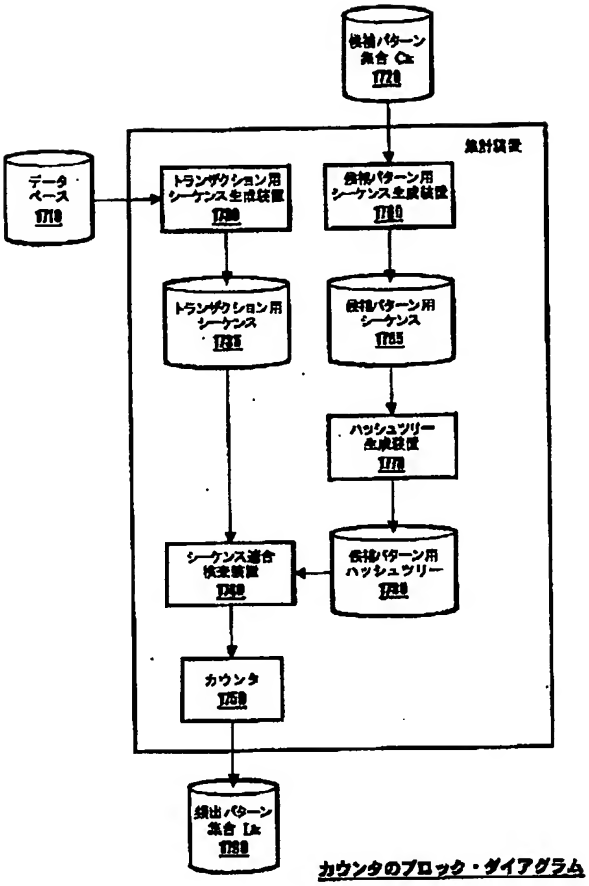
シーケンス・マッチング（２）

【図２５】

predicate	arguments
install	(customer, software)
get	(customer, fatal error, now)
push	(sbc, key)

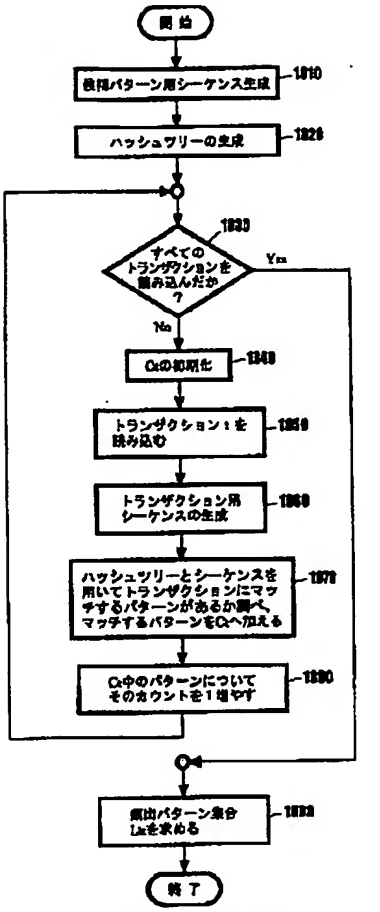
述語引数構造

【図１７】



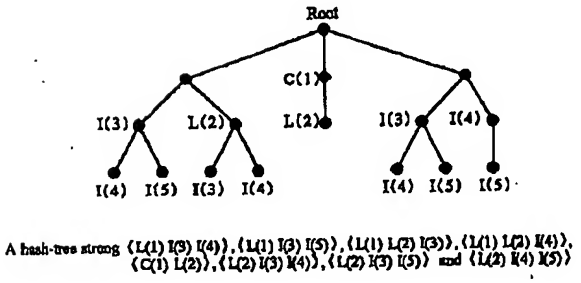
カウンタのブロック・ダイアグラム

【図１８】



２階層木構造データの統計方法のフローチャート

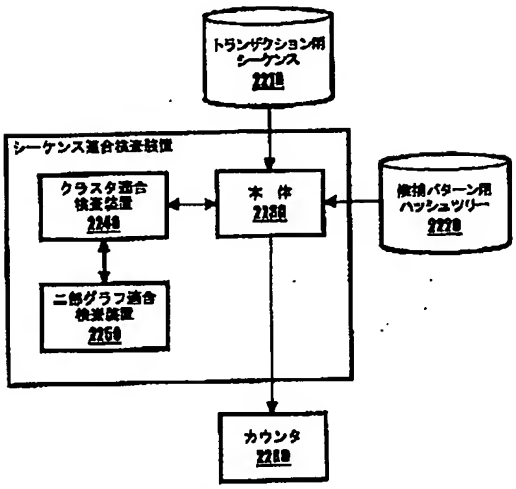
【図２０】



A hash-tree strong ⟨L(1) I(3) I(4)⟩, ⟨L(1) I(3) I(5)⟩, ⟨L(1) L(2) I(3)⟩, ⟨L(1) L(2) I(4)⟩, ⟨C(1) L(2)⟩, ⟨L(2) I(3) I(4)⟩, ⟨L(2) I(3) I(5)⟩ and ⟨L(2) I(4) I(5)⟩

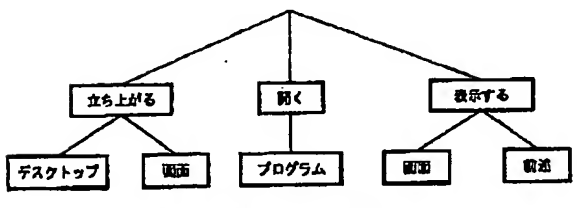
ハッシュ木

【図２２】



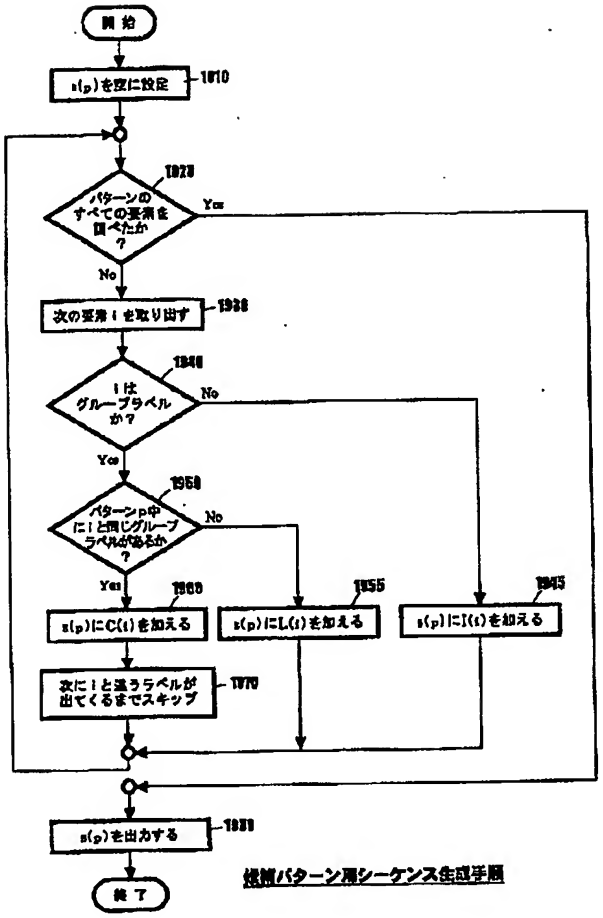
シーケンス適合のブロック・ダイアグラム

【図２７】



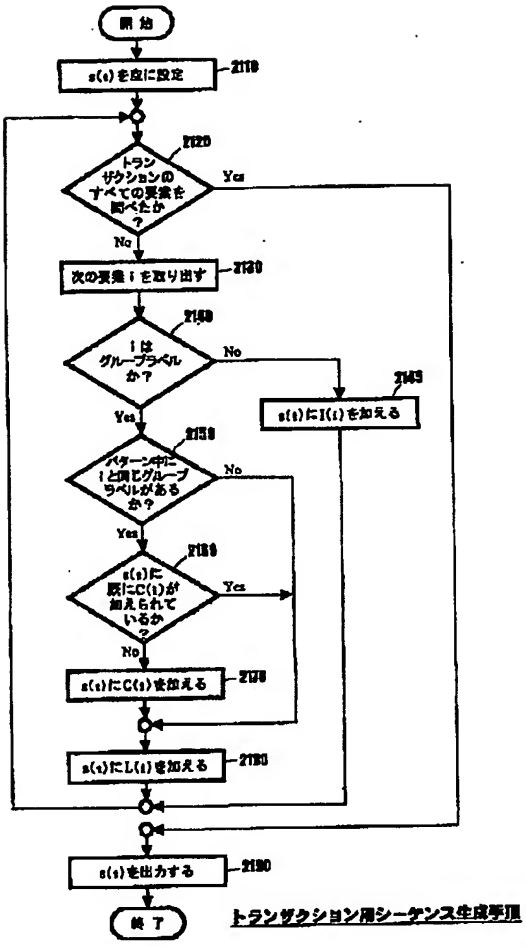
作成された構造化データ

【図１９】



候補パターン用シーケンス生成手順

【図２１】

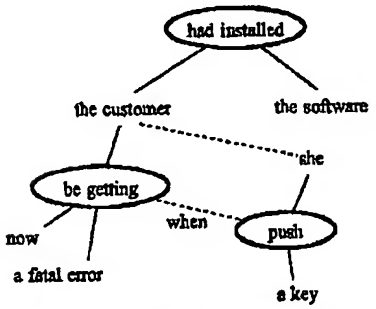


トランザクション用シーケンス生成手順

【図２４】

自然言語

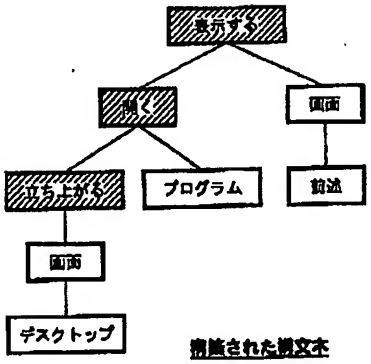
The customer had installed the software, and is now getting a fatal error when pushes a key.



Circled words are predicative verbs.

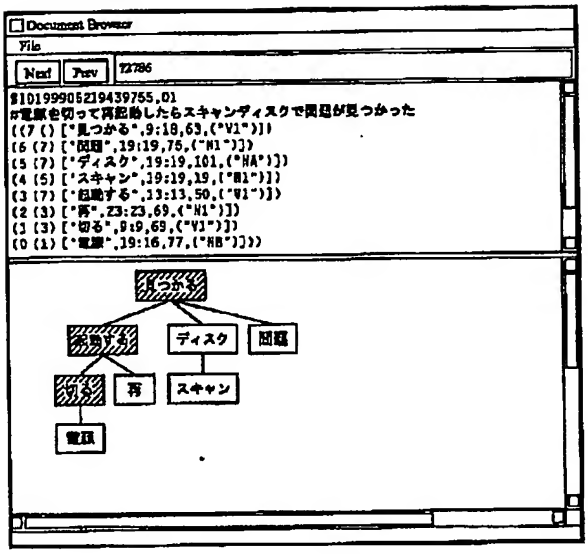
ネットワーク例

【図２６】

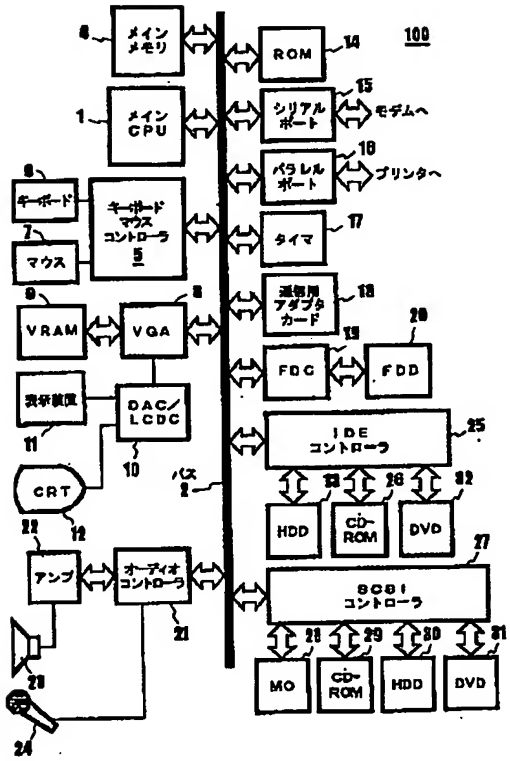


構築された構文木

【図２８】



【図２９】



フロントページの続き

(72)発明者　福田　剛志
神奈川県大和市下鶴間1623番地14　日本アイ・ビー・エム株式会社　東京基礎研究所内

(72)発明者　松澤　裕史
神奈川県大和市下鶴間1623番地14　日本アイ・ビー・エム株式会社　東京基礎研究所内

Ｆターム(参考)　5B075 ND03 NK02 NK32 NK43 NK45
NR03 NR12 PP25 PR04 UU06